VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Indexace sociálních sítí
Indexation of Social Networks

2011

Jakub Morcinek

„Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.“

V Ostravě 6.května 2011 ……

Na tomto místě bych chtěl poděkovat panu Ing.Petrovi Schererovi, který mi pomohl při psaní práce, bez jeho odborného vedení by tato práce nevznikla. Chtěl bych poděkovat také své rodině za podporu a trpělivost.

Abstrakt

Sociální síť je v dnešní době velmi oblíbenou metodou komunikace – především u mladých lidí.Bakalářská práce obsahuje stručný popis nejvíce oblíbených sítí a pak detailní zaměření na jednu z nich.Na tuto vybranou síť bude sestrojen crawler – robot, který bude procházet a indexovat obsah sítě.Práce obsahuje popis postupu crawlera od začátku až do posledního stupně zanoření sítě.Zjistilo se že crawler bude muset k sociální sítí přistupovat anonymně takže se bude používat proxy systémy. Použilo se různých metod práce s textem, vyhledávání v textu, ukládání textu.V práci nalezneme i stručný popis použití regulárních výrazů a jejich použití. Jeden s cílů práce je sestrojit graf tudíž se vytvoří menší seznam grafů a z nich pak vybere nejvhodnější pro zobrazení nasbíraných dat. Správný výběr grafu bude hrát důležitou roli v následném zobrazení.Graf by neměl být moc velký a měl by být přehledný.

Klíčová slova: Sociální síť; crawler; indexace; regulární výrazy;

Abstract

Social networking is today a very popular method of communication - especially among young people. Bachelor thesis contains a brief description of the most popular network and then specifically to one of them At the selected network is constructed crawler - a robot that will crawl and index the content network. The work contains a description of the crawler from the beginning to the last stage of the network plunge. It was found that the crawler will have access to social networks anonymously so it will use the proxy system. Used the different methods of working with text, text searching, storing text. In this work we find a brief description of the use of regular expressions and their use. One of our goals is to construct a graph thus creates a smaller list of charts and then select the best one to display collected data. Proper selection of the graph will play an important role in the subsequent view. Graph should not be too large and should be read.

Keywords: Social network; crawler; indexing; regular expressions;

Seznam použitých zkratek

MySQL – Databázový systém
SŘBD – Systém řízení báze dat
DB – Databáze
HTML - HyperText Markup Language
URL - Uniform Resource Locator

Obsah

Seznam obrázku

Seznam tabulek

Seznam výpisů zdrojového kódu

1. Úvod

V posledních letech náš život začaly ovlivňovat sociální sítě. Názory na tyto sítě se liší, někdo je miluje a pěje na ně ódy a jiní je zas zatracují a nemají je rádi. Úlohou mé práce bylo se zaměřit na jednu z nich a sestrojit crawlera, který by vybranou síť indexoval. Zaměřím se na sociální sítě obecně a v pozdějších kapitolách už jen na vybranou sociální síť.

V úvodních kapitolách si nejprve vysvětlím co to vlastně sociální síť je, jakou má definici a pak si zvolím jednu, kterou si podrobně nastuduji, seznámím se s ní a hlavně se seznámím se zdrojovým kódem sítě. Určitě bude vhodné nastudovat a připravit si vhodný systém přístupu na síť abych se vyhnul pozdějším problémům na sociální sítí.

Po úvodním seznámení se dostanu k samotné implementaci crawlera a k jeho jednotlivý pracovním úkonům od přístupu k sítí a po nejhlubší zanoření do sítě. Jelikož vyhledávání informací v textu bude crawler provádět pomocí regulárních výrazů věnuji i jim pár podkapitol abych osvětlil co jsou, k čemu jsou, jak se používají.

V závěru práce budu sestavovat grafy a statistiky s informací, které mi crawler nasbírá na sítí. Bude třeba se obeznámit s grafy a vybrat ten nejvhodnější – nejpřehlednější. Statistiku nebudu dělat z celého fóra, ale jen s určitých částí nebo časových intervalů.

2. Soc. sítě

2.1 Definice sociálních sítí

Sociální síť, zvaná též společenská síť, komunitní síť či komunita, anglicky social network, je propojená skupina lidí, kteří se navzájem ovlivňují. Sociální síť není tvořena na základě zájmů, vazeb nebo z podobných důvodů např. spolužáci ze školy apod. Jako komunitní sítě se označují tyto webové servery (toto je např. jasně řečeno i na internetových stránkách Facebooku, tento portál bývá často nesprávně označován jako sociální síť, přitom se jedná o typickou komunitní webovou prezentaci neboť soustřeďuje kamarády, známé, skupiny které mají společné zájmy, a nebo mají potřebu si předávat určité informace).[1]

Z názvů „sociální síť“, „společenská síť“, „komunitní síť“ a „komunita“ je nejfrekventovanější výraz sociální síť, z etymologického hlediska je ale nejsprávnější výraz společenská síť.

Pojem „sociální síť“ se dnes také často používá ve spojení s internetem a nástupem webů, které se na vytváření sociálních sítí přímo zaměřují (Seznamka.cz, Lidé.cz, Facebook Štěstí.cz…).

Zde je krátký seznam sociálních sítí a jejich stručný popis:

- **Facebook** – slouží jako sociální síť, internetová seznamka, herní server, pro internetové profily lidí, podniků i dalších subjektů, pro internetová fóra, pro ukládání a sdílení multimédií, nejpoužívanější sociální síť na světě
- **Myspace** – slouží jako sociální síť, pro internetové profily lidí, pro ukládání a sdílení multimédií, druhá nejpoužívanější sociální síť na světě a podle mnohých lidí je nejlepší sociální sítí na světě, patří společnosti News Corporation, kterou vlastní Rupert Murdoch
- **Twitter** – slouží především pro mikroblogy a jako sociální síť
- **Tuenti** – slouží jako sociální síť, přezdíváno „Španělským Facebookem“ (založen v Madridu)
- **Lidé.cz** – slouží jako sociální síť, pro internetová fóra, pro internetové profily především lidí, pro blogy, jako chatovací server, jako internetová seznamka, pro internetové kurzy, pro ukládání a sdílení fotografií, tento server je uživatelsky spojen se sociální sítí Spolužáci.cz

- **Spolužáci.cz** – slouží jako sociální síť, pro internetové profily, pro ukládání a sdílení multimédií, jako chatovací server, je určen pro spolužáky a bývalé spolužáky, tento server je uživatelsky spojen se serverem Lidé.cz, obdoba amerického Classmates
- **Badoo** – slouží jako sociální síť, pro internetové profily
- **Seznamka.cz** – slouží jako internetová seznamka, sociální síť
- **Štěstí.cz** – slouží jako internetová seznamka, sociální síť
- **Líbímseti.cz** – slouží pro internetové profily lidí, jako internetová seznamka, jako sociální síť, jako chatovací server, pro blogy
- **XChat.cz** – slouží jako chatovací server, pro internetové profily lidí, jako sociální síť, pro internetová fóra, jako herní server
- **LinkedIn** – slouží jako sociální síť a pro internetové profily a pro pracovní životopisy, je to pracovní sociální síť, mnohým světovým zaměstnavatelům stačí místo životopisu poslat odkaz na profil na LinkedIn, zajímavá je možnost vystavení referencí o podrobnostech spolupráce s kolegy z LinkedIn
- **Naymz** – slouží jako sociální síť a pro internetové profily a pro pracovní životopisy, je to pracovní sociální síť, mnohým světovým zaměstnavatelům stačí místo životopisu poslat odkaz na profil na Naymz
- **Orkut** – slouží jako sociální služba, jako chatovací server, pro ukládání a sdílení multimédií, zajímavostí je, že spadá pod Google, v Brazílii je nejpopulárnější internetovou stránkou
- **Hi5** – slouží jako sociální síť, podle společnosti comScore byla tato síť v roce 2008 třetí nejúspěšnější sociální sítí, co se týče počtu unikátních uživatelů za měsíc, na internetu na světě
- **Xing** – slouží jako sociální síť, je to pracovní sociální síť
- **Bebo**– slouží jako sociální síť, od roku 2008 patří společnosti AOL
- **Classmates** – slouží jako sociální síť, je určen pro spolužáky a bývalé spolužáky, jeho obdobou je český server Spolužáci.cz, od roku 2004 ho vlastní společnost United Online
- **Friendster** – slouží jako sociální síť
- **Blackplanet** – slouží jako sociální síť, je určen pro Afroameričany a jejich známé

2.2 ESET

Pro práci se nejlépe osvědčila síť Wilders Security Forum součástí této komunity zabývající se viry malwarem a ochranou proti těmto škůdcům.Toto fórum je v anglickém jazyce nezabývám se jím celým, ale pouze subfórem, které je vyznačeno pro firmu Eset a její produkty – antivirové programy NOD. Příspěvky zde jsou věcné a řeší problémy uživatelů, kteří si již nevědí rady nebo chtějí pomoct jiným s počítačovými viry. Po přečtení pár vláken určitých problémů jsem zjistil, že příspěvky se neopakují a je tady minimum příspěvků typu off topic(to jsou topicy které k tématu neříkají nic a správnými administrátory by měly být smazány). Firma ESET, která byla založena v roce 1992, je světovým producentem bezpečnostního software pro firemní klientelu i koncové uživatele a věnuje se boji proti vznikajícím počítačovým hrozbám. Produkty ESET patří mezi nejpokročilejší bezpečnostní softwarová řešení na světě, což je možné prokázat řadou prestižních ocenění.[2]

2.3 Vzhled fóra

Obrázek 1: Vzhled vybraného fóra

2.4 Fórum v číslech

Název sekce	Počet vláken cca	Počet příspěvků cca
ESET NOD32 Antivirus Forum	6,904	49,355
ESET Smart Security Forum	4,485	28,147
ESET Beta Support Forum	147	1,064
NOD32 v2 Antivirus Forum	9,544	80785
Other ESET Products Forum	1,072	4,525

Tabulka 1: Orientační počet vláken a příspěvku fóra

3. Analýza

3.1 Použité Technologie

MySql – MySQL je multiplatformní SŘDB. Komunikace s ní probíhá – jak už název napovídá – pomocí jazyka SQL. Podobně jako u ostatních SQL databází se jedná o dialekt tohoto jazyka s některými rozšířeními. Pro svou snadnou implementovatelnost (lze jej instalovat na Linux, MS Windows, ale i další operační systémy), výkon a především díky tomu, že se jedná o volně šiřitelný software, má vysoký podíl na v současné době používaných databázích. MySQL bylo od počátku optimalizováno především na rychlost, a to i za cenu některých zjednodušení: má jen jednoduché způsoby zálohování, a až donedávna nepodporovalo pohledy, triggery, a uložené procedury. Tyto vlastnosti jsou doplňovány teprve v posledních letech [3]. Bude to základní stavební kámen mého crawleru do databáze se budou ukládat všechny informace které posbírám.

Java – Díky své přenositelnosti je používán pro programy, které mají pracovat na různých systémech počínaje čipovými kartami (platforma JavaCard), přes mobilní telefony a různá zabudovaná zařízení (platforma Java ME), aplikace pro desktopové počítače (platforma Java SE) až po rozsáhlé distribuované systémy pracující na řadě spolupracujících počítačů rozprostřené po celém světě (platforma Java EE). Tyto technologie se jako celek nazývají platforma Java . [4]Tento programovací

jazyk jsem si vybral, protože mi nejvíc sedí a mám s ním nejvíce zkušenosti. Pomocí něj budu otevírat URL a sbírat informace.

Proxy list – Proxy server funguje jako prostředník mezi klientem a cílovým počítačem (serverem), překládá klientské požadavky a vůči cílovému počítači vystupuje sám jako klient. Přijatou odpověď následně odesílá zpět na klienta. Může se jednat jak o specializovaný hardware, tak o software provozovaný na běžném počítači.[4] Aby mě s fora neodstranili a nezabanovali musím si měnit ip adresy toho docílím právě proxy a důmyslným výběrem proxy. [4]

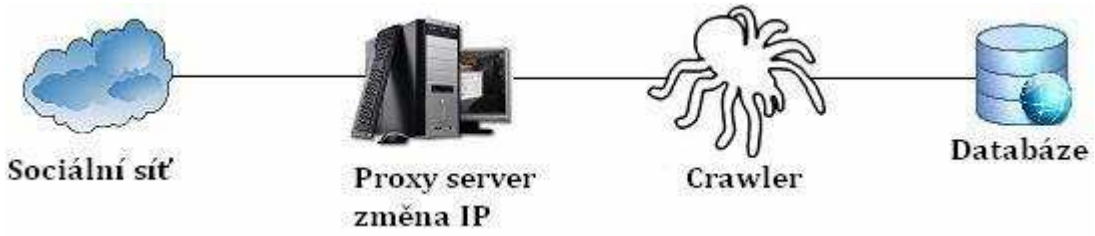
Existují dva druhy proxylistu, veřejný a soukromý. Rozdíl mezi nimi je prostý – soukromý proxylist je rychlejší a spolehlivější, kdyžto ten veřejný má problémy jak s chodem tak přenosovou rychlostí. Z mých zkušeností se zdá že najít spolehlivé proxy na víc než jeden nebo dva dny je celkem náročný úkol.

3.2 Webcrawler

Crawler nebo taky Web Crawler – Program běžící na počítači, který automaticky prochází internet, nebo jednotlivé stránky. Je to typ robota nebo softwarového agenta, který má na začátku seznam URL, které má navštívit, v našem případě jen jednu a to základní adresu fóra. Jakmile crawler navštíví toto URL vyhledá hypertextové odkazy, které jsou s tímto URL svázány, ty pak přidá na seznam míst které má navštívit a ty pak prochází. Crawleři mohou být použiti pro automatizovanou práci na webu např.validace HTML kódu nebo automatická údržba webu. Mnohem častěji jsou však zneužívání ke sběru více specifických informací např.emailové adresy->spam.Crawler sestrojený v této práci bude sbírat informace o příspěvcích, jejich četnosti, jejich autorech … .

3.3 Návrh implementace

Celá práce by měla proběhnout takto:



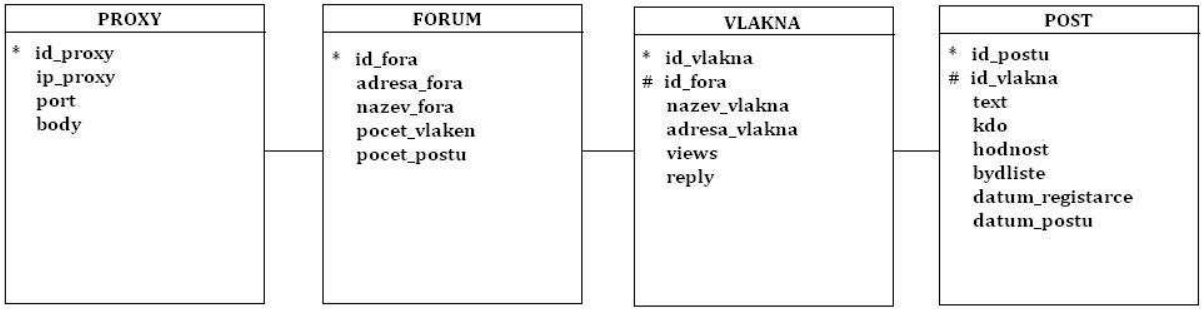
Obrázek 2: Postup práce

Informace ze sociální sítě povedou přes proxy server (kvůli změně IP) ke crawlerovi a ten až zpracuje, vybere informace, které mě zajímají. Provede uložení do dababáze.

Základní prvek práce je databáze bude obsahovat 4 tabulky:

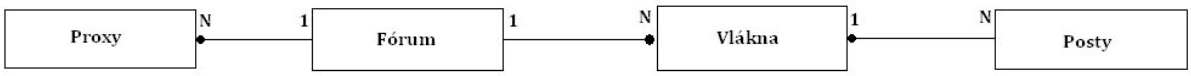
1)Proxy list
2)Seznam fór
3)Seznam vláken v jednom fóru
4)Seznam příspěvku v jednom vláknu

Konceptuální schéma databáze:



Obrázek 3: Konceptuální schéma databáze

ER Diagram:



Obrázek 4: ER Diagram

Datový slovník:

	datový typ	délka	key	null	význam
FORUM					
id_fora	int	11	P	NE	Unikátní
adresa_fora	varchar	300	NE	ANO	pro pozdější vnořování do fora
nazev_fora	varchar	300	NE	ANO	také pro budoucí vnoření
pocet_vlaken	int	11	NE	ANO	pro kontrolu stavu
pocet_postu	int	11	NE	ANO	pro kontrolu stavu
VLAKNA					
id_vlakna	int	11	P	NE	unikátní číslo –auto_increment
cislo_fora	int	11	F	NE	cizí klíč tabulky forum – id_fora
adresa_vlakna	varchar	300	NE	ANO	pro pozdější vnoření
nazev_vlakna	varchar	300	NE	ANO	pro pozdější vnoření
views	int	11	NE	ANO	pro statistiku
reply	int	11	NE	ANO	pro kontrolu stavu
pocet_stranek	int	11	NE	ANO	pro procházení stránek fora
POST					
id_postu	int	11	P	NE	unikátní číslo – auto_incement
id_vlakna	int	11	F	NE	cizí klíč tabulky vlákna
kdo	varchar	300	NE	ANO	pro statistiku
datum_postu	datetime	-	NE	ANO	pro statistiku
datum_registrace	date	-	NE	ANO	pro statistiku
hodnost	varchar	300	NE	ANO	pro statistiku
text	longText	-	NE	ANO	pro statistiku
bydliste	varchar	100	NE	ANO	pro statistiku
PROXY					

id_proxy	int	11	P	NE unikátní číslo – auto_incement
ip_proxy	varchar	100	NE	ANO parametr proxy
port	int	11	NE	ANO parametr proxy
body	int	3	NE	ANO proxy body

V tomto projektu je třeba se zaměřit na práci s textem – provést selekci určité části a ostatní si ponechat v záloze. Bylo by velice neefektivní kdyby se text procházel hrubou silou a hledala by se určitá po sobě jdoucí shoda. Efektivní metodou jak vyhledat v textu určitý výraz je použití regulárních výrazů.

3.4 Regulární výrazy

Jako v aritmetice můžeme pomocí operátorů + a * vytvářet výrazy jako (1+2)*3 můžeme v teorii formálních formálních jazyků pomocí operátorů +, · a * vytvářet tzv. regulární výrazy, jako třeba (0 + 1) · 0 *. Jako je hodnotou aritmetického výrazu (5 + 3) × 4 číslo 32, je hodnotou regulárního výrazu (0 + 1) · 0* jazyk ({0} U {1}) · {0}*.[6]

Induktivní definice regulárních výrazů nad abecedou ∑ :

Ø, Ɛ, a (kde a ∑) jsou regulární výrazy:
Ø . . . označuje prázdný jazyk
Ɛ . . . označuje jazyk {Ɛ}
a . . . označuje jazyk {a}

Jestliže α, β jsou regulární výrazy, pak i (α + β), (α · β), (α*) jsou regulární výrazy:
(α + β). . . označuje sjednoceni jazyků označených α a β
(α · β). . . označuje zřetězeni jazyků označených α a β
(α*). . … označuje iteraci jazyka označených αú7Neexistuji žádné další regulární výrazy než ty definované podle předchozích dvou bodů. [6]

Podle definice jsou 0 i 1 regulární výrazy.
Protože 0 i 1 jsou regulární výrazy, je i (0 +1) regulární výraz.
Protože 0 je regulární vyraz, je i (0*) regulární výraz.
Protože (0 +1) i (0*) jsou regulární výrazy, je i ((0 +1) · (0*)) regulární výraz.

Jestliže α je regulární výraz, zápisem [α] označujeme jazyk definovaný regulárním výrazem α .

[((0 +1) · (0*))] = {0, 1, 00, 10, 000, 100, 0000, 1000, 00000, . . . }[6]

3.5 Zápis regulárních výrazů

Aby byl zápis regulárních výrazů přehlednější a stručnější, používáme následuji pravidla: [6]

1)Vynecháváme vnější par závorek.
2)Vynecháváme závorky, které jsou zbytečné vzhledem k asociativitě operaci sjednoceni (+) a zřetězeni (·).
3)Vynecháváme závorky, které jsou zbytečné vzhledem k prioritě operaci (nejvyšší prioritu má iterace (*), menši zřetězeni (·) a nejmenší sjednoceni (+)).
4)Nepíšeme tečku pro zřetězeni.

Příklad: místo

(((((0 · 1)*) · 1) · (1 · 1)) + (((0 · 0) + 1)*))

obvykle píšeme

(01)*111 + (00 +1)*

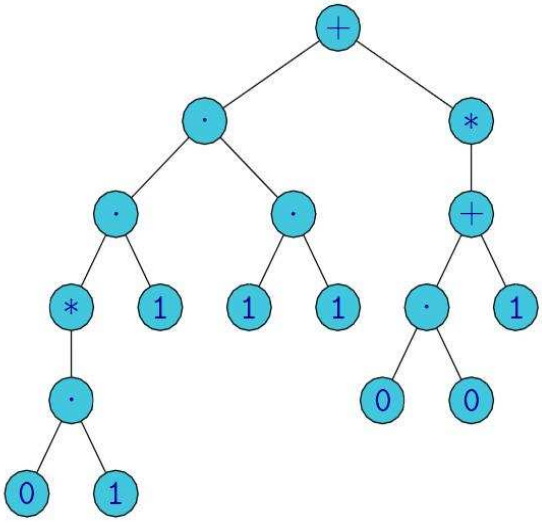
3.6 Příklady regulárních výrazů

Ve všech případech ∑ = {0, 1}.
0 . . . jazyk tvořeny jediným slovem 0
01 . . . jazyk tvořeny jediným slovem 01
0 + 1 . . . jazyk tvořeny dvěma slovy 0 a 1
0∗ . . . jazyk tvořeny slovy ", 0, 00, 000, . . .
(01)∗ . . . jazyk tvořeny slovy ", 01, 0101, 010101, . . .
(0 +1)∗ . . . jazyk tvořeny všemi slovy nad abecedou {0, 1}
(0 +1)∗00 . . . jazyk tvořeny všemi slovy končícími 00
(01)∗111(01)∗ . . . jazyk tvořeny všemi slovy obsahujícími podslovo 111
předcházené i následované libovolným počtem slov 01

(0 +1)∗00 + (01)∗111(01)∗ . . . jazyk tvořeny všemi slovy, která buď končí 00 nebo obsahují podslovo 111 předcházené i následované libovolným počtem slov 01
(0 +1)∗1(0+1)∗ . . . jazyk tvořeny všemi slovy obsahujícími alespoň jeden symbol 1
0∗(10∗10∗)∗ . . . jazyk tvořeny všemi slovy obsahujícími sudy počet symbolů 1[6]

3.7 Graf regulárního výrazu

Strukturu regulárního výrazu si můžeme znázornit jako strom[6]:



$$(((((0 \cdot 1)^*) \cdot 1) \cdot (1 \cdot 1)) + (((0 \cdot 0) + 1)^*))$$

Obrázek 5: Graf regulárního výrazu

3.8 Průběh indexace

První tabulka bude ukládat seznam jednotlivých fór. Při první návštěvě fóra se do ni uloží, a při každé další návštěvě se bude kontrolovat jejich počet. Pokud přibude další fórum uloží se k těm ostatním. Pokud se nějaké smaže bude třeba před smazáním zjistit o které šlo a pak tomu fóru zadat příznak neaktivní, aby crawler na toto fórum dal nedotazoval a tím ty dotazy šetřil.

Dále pak v rámci úspor dotazů na server budu ke každému fóru ukládat počet aktuálních příspěvků. Při každé návštěvě počínaje druhou se budou tyto informace kontrolovat, jestli počet příspěvků bude totožný, nebudu toto fórum procházet a ušetřím tak spoustu dotazů.

Po tom co kód celé stránky projde přes regulární výrazy dostanu jen jeden řetězec, který budu ukládat do databáze. Řetězec budu muset rozdělit do několika části tak, aby se mi s ním dobře pracovalo a data z databáze se pak vytahovaly dobře to pro mě znamená, že nebudu celý řetězec dělit na straně javy, ale na úrovni databáze ze, které si pak jednotlivé části budu vytahovat.

Po uložení všech fór bude potřeba zanořit se o úroveň níže a uložit jednotlivé vlákna daného fóra.Vlákna daného fóra se budou ukládat do databáze s cizím klíčem nadřazeného fóra takže si zpětně budu moct dohledat ke kterému patří a toto platí i pro jednotlivé příspěvky, které budou mít taky cizí klíče, ale už ne fór, ale jednotlivých vláken takže bude zjistitelné který příspěvek patří jakému vláknu a fóru – hezky tím zůstane zachována struktura fóra a zároveň se bude dobře tvořit graf fóra.

Po vnoření o jednu úroveň se jednotlivá vlákna nechají zpracovat přes regulární výrazy a dostanu znovu řetězec, který si podělím na určité části a uložím do databáze. Při ponoření o další úroveň se dostaneme k samotnému příspěvku a ten se uloží se všemi možnými informacemi kvůli rozmanitosti statistik, které později budou prováděny právě na základě uložených příspěvků

4. Implementace

4.1 Proxy systém

Proxy list bude fungovat na bázi kladných bodů. Každá proxy bude začínat s určitým obnosem bodů. Před její použitím se bude kontrolovat dotazem na Google.cz zda je proxy funkční a je možno jí použít. Jestli se vrátí kladná odezva v podobě odpovědí 200 tak se tato proxy použije a přičte se jí kladný bod. Pokud se vrátí Bad request 400 nebo se nevrátí nic bude se losovat jiná proxy.
Nejprve je třeba vytáhnout si pomocí SQL dotazu parametry proxy. Proxy se vytahuje z databáze náhodným způsobem:

```
String dotaz = "SELECT * FROM proxy WHERE id_proxy = ?";
```

Výpis 1: Získání informací o proxy

Výpis 1 znázorňuje dotaz při kterém získám informace o proxy, kterou chci použít.

```
proxy = new Proxy(Proxy.Type.HTTP,
        new InetSocketAddress(proxyna.getIpProxy(),proxyna.getPortProxy()));
```

Výpis 2: Příprava nastavení proxyny

Na výpisu 2 je vidět přípravu získaných hodnot na adresu kterou chci otevřít.

```
HttpURLConnection urlcon = (HttpURLConnection).cestaKforu.openConnection(proxy);
```

Výpis 3: Aplikace proxy na URL které chci otevřít

Samotnou aplikaci proxy je vidět na výpisu 3.

```
if(responseCode == 200){
        jeOukej = true;
        body++;
        proxyna.setBody(body);
```

```
        work.update(proxyna);
      }else{
        body--;
        proxyna.setBody(body);
        work.update(proxyna);
      }
```

Výpis 4: Úprava bodů jednotlivých proxy

Na výpise číslo 4. je vidět co se stane pokud proxy projde nebo neprojde testem. Body jsou upravovány ihned v případě úspěchu je proxy hned poslána dál.

Před každým spuštěním se bude proxy list ověřovat na „mrtvé proxy“ provede se to jednoduše dotazem SQL, který vyřadí všechny proxy, které mají body 0

```
String smazProxy = "DELETE  FROM proxy WHERE body = "0" ;
```

Výpis 5: Odstranění „mrtvé proxy“

4.2 Spojení javy s MySQL

Java používá pro přístup k databázím speciální ovladač, který se jmenuje JDBC. Je to takové příjemné prostředí, pomocí kterého se můžete připojit prakticky k jakékoli databázi. Poskytuje základní rozhraní pro unifikovaný přístup k databázím, díky tomuto se můžete naučit jednotný přístup k databázím, nemusíte tak znát jemné niance jednotlivých druhů databází.

Existují 4 druhy JDBC rozhraní, my se budeme pouze zabývat možnostmi připojení se k databázi MySQL. Pro každý druh databáze poskytuje jejich výrobce své JDBC rozhraní.[5]

4.3 Vnoření do prvního stupně sociální sítě

Pokud proxy uspěje v testu použijeme ji na otevření adresy sociální sítě

```
http://www.wilderssecurity.com/forumdisplay.php?f=15
```

Výpis 6: URL sociální sítě

a dostaneme HTML kód stránky, s toho kódu musíme vybrat informace, které nás zajímají a ty uložit do databáze. Zde najdou použití regulární výrazy. Nejdříve pomocí regulárního výrazu vyhledáme všechny hypertextové odkazy, jelikož vnoření do další úrovně sociální sítě se provádí přes ně. Na to si vytvoříme vhodný výraz.

```
Pattern p = Pattern.compile("<a (.*?)</a>");
Matcher m = p.matcher(inputLine);
```

Výpis 7: Použití regulárního výrazu

Ve výpisu číslo 7 je vidět jednoduché, ale účinné použití regulárního výrazu do patternu si zvolím výraz, který hledám a ten pattern pak použiji v našem případě na inputLine ve kterém je uložen HTML kód celé stránky.

Nyní budeme pracovat již s tímto výběrem z textu a ten musí obsahovat HTML tag <a.Nad tímto výběrem je třeba sestrojit další filtr. Když text projde tímto filtrem dostaneme na výstup název fóra.V proměnné group(0) je uložen výběr textu, který se vygeneroval po použití regulárního výrazu výše.

```
String [] pole = m.group(0).split("s=");
      if(pole.length == 2){
```

```
druheKolo = pole[1].split("f=");
Pattern tretiKolo = Pattern.compile("<(\\S+?).*?>(.*?)</\\1>");
try{
    strong = tretiKolo.matcher(druheKolo[1]);
   while(strong.find()) {
    try{
     nazevFora = strong.group(2);
```

Výpis 8: Filtr sestrojený pomocí regulárních výrazů

K názvu fóra je zapotřebí zjistit číslo do HTML odkazu, který je zapotřebí pro další vnoření. Zde použijeme regulární výraz, který nám najde všechny čísla, které máme ve vyfiltrovaném textovém výřezu.

```
Pattern ctvrteKolo = Pattern.compile("^[0-9]*");
    try{
      cislo = ctvrteKolo.matcher(druheKolo[1]);
        while(cislo.find()) {
          try{
            cislo2 = cislo.group(0);
          }catch(Exception e){
              e.printStackTrace();
```

Výpis 9: Filtr na získání čísla vlákna

Po takto vyfiltrovaném textu si složím celý link, který dále budu potřebovat pro další práci.

```
String link = pole[0]+"f=" + cislo2 ;
```

Výpis 10: Sestrojení URL jednotlivých sekcí

Chybí zjisti kolik vláken a postů fórum toto číslo lze najít v <td> s classem alt1 a alt2 je třeba tyto buňky najít a projít jejich obsah.

```
if(inputLine.contains("alt1")) {
        if(0 != new Parser().vratPocet(inputLine)) {
                sekce.setVlakna_fora(new Parser().vratPocet(inputLine));
        }

}
```

Výpis 11: Filtr na získání počtů vláken a postů

Kód z výpisu 11 nám vyfiltruje pouze buňky s classem alt1 a dále třída Parser najde čísla, která potřebuji.

```
Pattern altcisla = Pattern.compile(">[0-9]*((,[0-9]*)|[0-9])<");
    Matcher ma = altcisla.matcher(alt);
    String pocetCiselBezZobaku = "";
      while(ma.find()) {
          pocetCiselBezZobaku = ma.group(0).replaceAll(">", "");
          pocetCiselBezZobaku = pocetCiselBezZobaku.replaceAll("<", "");
          pocetCiselBezZobaku = pocetCiselBezZobaku.replaceAll(",", "");
          pocet = Integer.parseInt(pocetCiselBezZobaku);
      }
return pocet;
```

Výpis 12: Odebrání nepotřebných znaků

Pomocí regulárního výrazu (">[0-9]*((,[0-9]*)|[0-9])<"); najdu všechny čísla, která jsou z obou stran osazena znaky > <.Chybí pouze se těchto znaků zbavit.Tato operace se provádí ve while, kde jsou nahrazovány. Postup pro zjištění buněk alt2 je podobný:

```
if(inputLine.contains("alt2")) {
        if(0 != new Parser().vratPocet(inputLine)) {
            sekce.setPosty_fora(new Parser().vratPocet(inputLine));
        }
}
```

Výpis 13: Naznačení filtru pro alt2

Pouze se změní podmínka na začátku, ale třída Parser zůstane nepozměněná. Zároveň v podmínkách nastavuji setry a jmenovitě Vlákna_fora od alt1 a Posty_fora od alt2. Po získání všech potřebných údajů můžu přikročit k uložení do databáze.

```
String dotaz = "INSERT INTO forum
(adresa_fora,nazev_fora,pocet_vlaken,pocet_postu,posledni_stranka) VALUES(?,?,?,?)";
PreparedStatement p = conn.getConn().prepareStatement(dotaz);
```

Výpis 14: Ukázka insertu do db

Při dalším spuštění crawlera je třeba ověřit jestli nevzniklo nějaké nové fórum to zajistí tato podmínka.

```
if(projizdec.hledejFora(ADRESA_FORA).values().size() == work.selectAdresaFora().size()) {
        work.smazThready();
        projetThready();
}
```

Výpis 15: Kontrola množství fór

4.4 *Vnoření do druhého stupně sociální sítě*

Pokud jsem se zanořil v sociální sítí o úroveň níž (k vláknům) je třeba vytvořit nový filtr, který projde jednotlivé stránky ve fóru do kterého jsem se zanořil.Takže je třeba zjisti kolik stránek celkově mají jednotlivá fóra. Tento problém vyřeším použitím regulárního výrazu.

```
Pattern najdiStranku = Pattern.compile("Page 1 of [0-9]* ");
Matcher posledniStranka = najdiStranku.matcher(inputLine);
```

Výpis 16: Nalezení poslední stránky

Poslední stránku lze v kódu nalézt ve frázi Page 1 of … právě na tuto frázi zaměřím svůj regulární výraz Page 1 of [0-9]* . Takto získanou poslední stránku si uložím a vnořené forum budu procházet pomocí inkrementování 1 v odkazu na další stránku do doby než se mnou získané číslo stránky bude rovnat stránce v odkazu
http://www.wilderssecurity.com/forumdisplay.php?f=88&**page=2**&order=desc
page=2 je druhou stránkou fóra ve kterém jsem a toto číslo budu zvyšovat.

Po vnoření do druhé úrovně si budu muset znova vytáhnout z celého HTML kódu stránky pouze ty pasáže, které mě zajímají. Použiji proto regulární výraz.

```
Pattern najdiLink = Pattern.compile("<a href=\"showthread.php?(.*?)</a>");
Matcher nalezeno = najdiLink.matcher(inputLine);
```

Výpis 17: Nalezení adres vláken

Tento výraz mi vytáhne hypertextové odkazy, které obsahují řetězec showthread.php.Tento řetězec obsahuje každý hypertextový odkaz vlákna a když si z něj vytáhnu pouze číslo, které mi pomůže zanořit se níž dostanu celou adresu fora.

```
rozdelLink = nalezeno.group(0).split(„t=“);
```

Výpis 18: Filtr pro čísla vláken

V nalezeno.group(0) mám vytříděný text a ten si funkci split rozdělím podle t= protože vím, že za tím je číslo pro mou potřebu.

```
Pattern najdiCislo = Pattern.compile("^[0-9]*");
Matcher hledaneCislo = najdiCislo.matcher(rozdelLink[1]);
```

Výpis 19: Filtr na nalezení čísel

Ve výpisu 19 je použit regulární výraz, který mi vyhledá potřebná čísla.

```
cisloVlakna = hledaneCislo.group(0);
thread.setAdresa_vlakna("http://www.wilderssecurity.com/showthread.php?t=" + cisloVlakna);
```

Výpis 20: Sestrojení URL

Tato část adresy vlákna se nebude měnit http://www.wilderssecurity.com/showthread.php?t= tudíž mi stačí když na konec za = přidám moje číslo, které jsem před chvílí dostal a mám kompletní adresu vlákna a když už ji mám nastavím si setr.

Dále musím zjistit název vlákna – regulárním výrazem. Zjistil jsem, že každý název je součástí tagu <a> a před ním je id css stylu čehož využiji a regulárním výrazem si najdu právě tyto výskyty.

```
Pattern druhej = Pattern.compile("thread_title_(.*?)</a>");
Matcher ma = druhej.matcher(inputLine);
```

Výpis 21: Nalezení názvů vláken

Tímto dostanu název vlákna, které je z obou stran sevřeno znaky >< čehož se zbavím použitím dalšího regulárního výrazu.

```
nazevVlakna = ma.group(0);
Pattern nazvyVlaken = Pattern.compile(">(.*?)<");
Matcher hotovyNazev = nazvyVlaken.matcher(nazevVlakna);
```

Výpis 22: Nalezení názvů vláken

V ma.group(0) mám uložený vyfiltrovaný řetězec a na něj použiji regulární výraz >(.*?)< a ten mi vyhledá názvy všech vláken na stránce. Ještě je třeba zjisti kolik odpovědí(replies) vlákno má a kolikrát bylo shlédnuto(views). Stačí najít řádek, který obsahuje určené slovo.

```
if(inputLine.contains("Replies:")) {
                int [] pocty = new Parser().vratPocetProThread(inputLine);
                replies = pocty[0];
                views = pocty[1];
}
```

Výpis 23: Nalezení řádku s potřebnými údaji

Podmínka mi vyhledá pouze řádky, které chci a pro tento případ jsem vytvořil instanci třídy Parser vratPocetProThread, která mi vrátí čísla která chci. Jelikož jsou oba čísla na jednom řádku je třeba zjistit, kdy začíná jedno a kdy druhé číslo.

```
String StringNaParsovani = alt.replaceAll(„alt2“, „QQ“);
      int pozice = StringNaParsovani.indexOf(„R“);
      int pozice = StringNaParsovani.indexOf(„V“);
      int poziceZobaku = StringNaParsovani.indexOf(„>“);
```

Výpis 24: Zjištění pozice jednotlivých písmen

:

Dynamicky si zjistím pozici R což je první písmeno v Replies a od toho místa provedu výběr řetězce pomocí substringu až do pozice V což je první písmeno ve slově Views a pokračuji do znaku > který řádek ukončuje.

```
String replies = StringNaParsovani.substring(poziceR,poziceV);
      replies = replies.replaceAll(",", "");
      replies = replies.replaceAll("Replies:","");
      replies = replies.trim();

a pro views

String views = StringNaParsovani.substring(poziceV,poziceZobaku);
      views = views.replaceAll("\"", "");
      views = views.replaceAll(",", "");
      views = views.replaceAll("Views:","");
      views = views.trim();
```

Výpis 25: Filtr na potřebná čísla Replies a Views

4.5 Vnoření do třetího stupně sociální sítě

Nyní jsem se dostal do posledního stupně zanořen. Tady najdu autory jednotlivých příspěvků, jejich text čas vložení a jiné drobnosti, které použiji do statistik. Nejprve jsem se rozhodnul najít si čas příspěvku toho dosáhnu podle patternu. Je důležité si pattern dobře poskládat, protože existuje více formátu času a tak je třeba si vyhledat co který symbol znamená a složit příslušny tvar.Pozor casesensitive to znamená že M neznamená to samé co m.

```
String pattern = "MMMM dd yyyy, hh:mm a";
SimpleDateFormat format = new SimpleDateFormat(pattern,Locale.ENGLISH);
```

Výpis 26: Filtr na data vložení příspěvku

Pak jsem si vytáhnul text příspěvku pomocí.

```
Pattern title = Pattern.compile(„<!—message →(.*?)<!-- / message →“);
Matcher postTitle = title.matcher(line.toString());
      while(postTitle.find()) {
            cistota =  postTitle.group().replaceAll(„<(.|\n)+?>“, „“).trim();
           text.add(cistota);
```

Výpis 27: Vytažení textu příspěvku

Dále potřebuji zjistit autora příspěvku.

```
Pattern anonym = Pattern.compile(„<div id=\“postmenu_[0-9]*\“>(.*?)<!-- / user info →“);
Matcher anonymFind = anonym.matcher(line.toString());
```

Výpis 28: Zjištění autora příspěvku

Jestli zadal bydliště tak zjistím i to .

```
try{
        String [] local = w.split("Location:");
        String [] cast = local [1].split("Posts:");
        home = cast[0].trim();
        bydliste.add(home);
}catch(Exception e){
        bydliste.add("NEZADANO");
}
```

Výpis 29: Informace o bydlišti autora

Z výpisu 29 je patrné, že autor příspěvku nemusí zadat bydliště. Pak je třeba s tím počítat a připravit se na to. Dále jestli bylo zadáno datum, kdy se připojil ke komunitě.

```
try{
        String [] z = w.split("Join Date:");
       save = z[1].substring(0,9);
       regDatum.add(save);
}catch(Exception e){
        regDatum.add("NEZADANO");
}
```

Výpis 30: Zjištění Join datu

Ve výpisu 30 se zase počítá s tím, že autor vlastně není členem tohoto fóra – což se v některých případech stává jde o takzvaného anonymního uživatele.

Tyto informace i ukládám do hash map a z nich posílám do databáze.

```
for(int i = 0; i < autori.size();i++) {
            post = new Posts();
            user = new User();
            user.setKdo(autori.get(i));
            user.setHodnost(hodnosti.get(i));
            user.setJoinDate(regDatum.get(i));
            user.setBydliste(bydliste.get(i));
            post.setText(text.get(i));
            post.setUser(user);
            post.setId_vlakna(thread.getId_vlakna());
            posty.add(post);
          }
```

Výpis 31: Plnění hash map

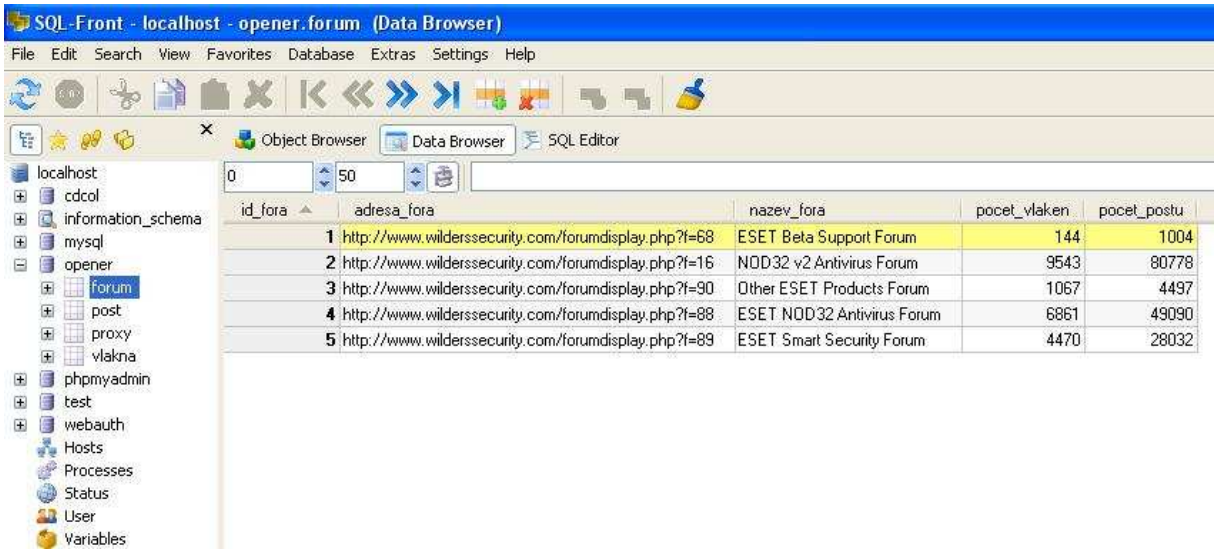
Výpis 31 ukazuje naplnění hash map cyklem for. Každý příspěvek má autora, tudíž cyklus for plní mapy dokud na stránce bude možno nalézt autory – i anonym je autor.

```
for(int i = 0; i < datumy.size();i++) {
            posty.get(i).setDatum(datumy.get(i));
            work.insertPost(posty.get(i),thread.getId_vlakna());
}
```

Výpis 32: Insert do databáze

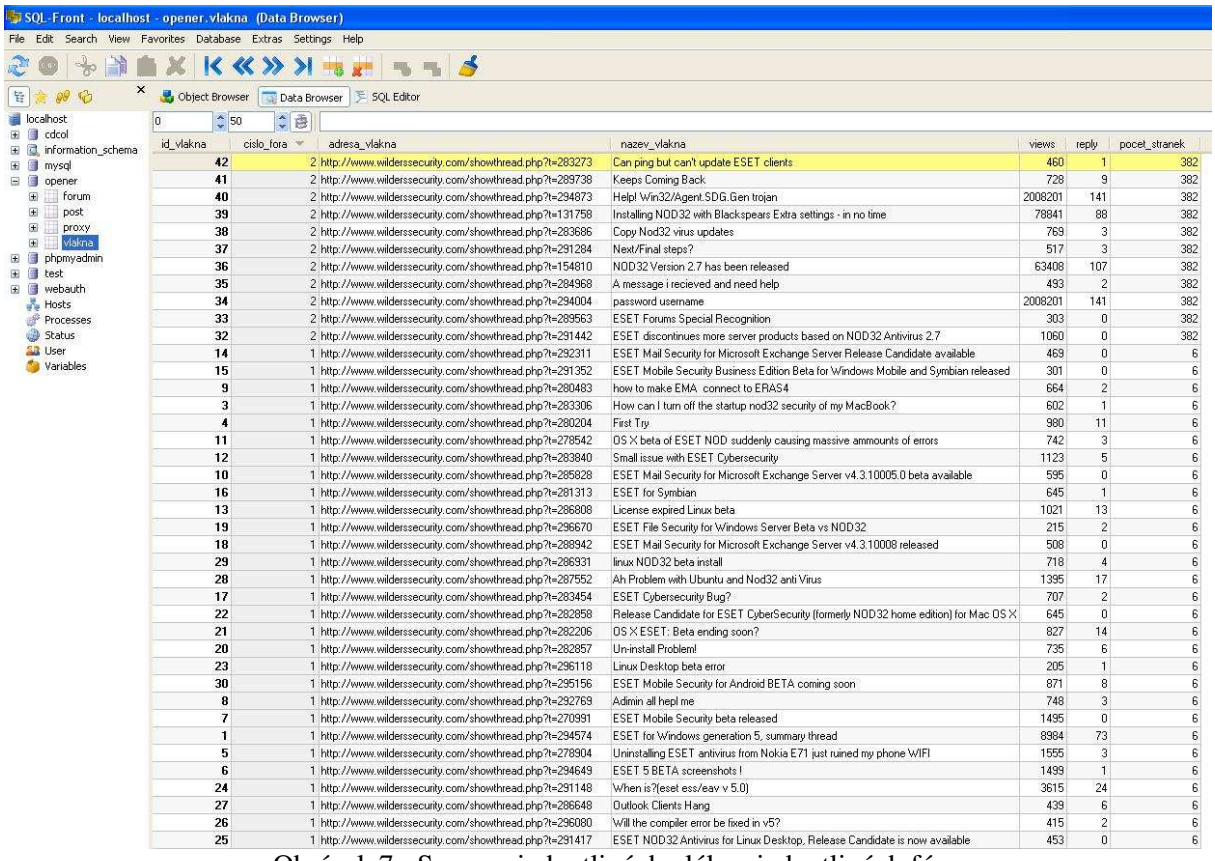
4.6 Ukázky výstupu ESET crawlera

.



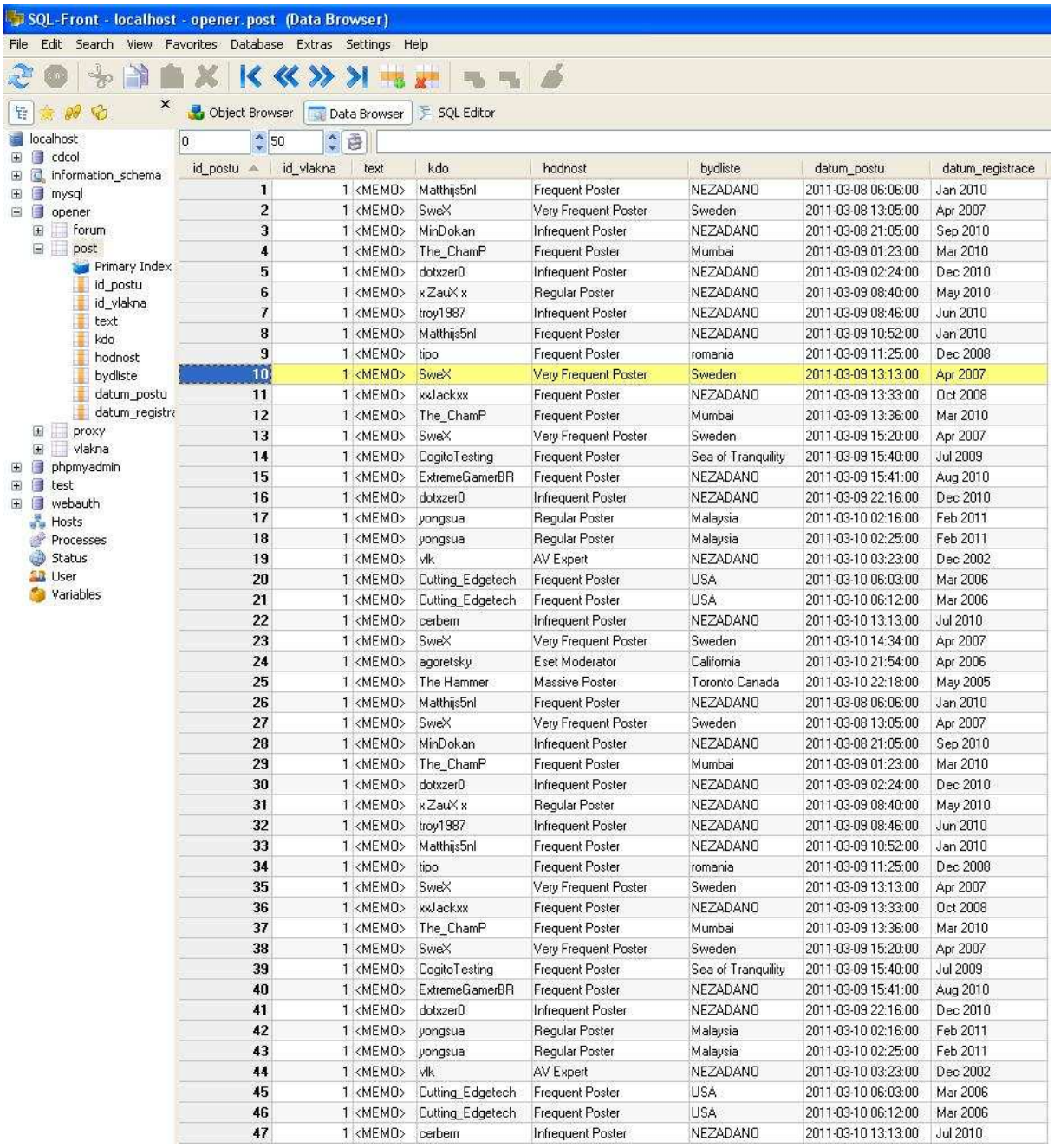
Obrázek 6: Seznam fór Esetu

Na obrázku číslo 6 můžeme vidět práci odvedenou crawlerem při prvním zanoření do sociální sítě.



Obrázek 7: Seznam jednotlivých vláken jednotlivých fór

Na obrázku 7 je znázorněn výsledek zanoření crawlera o úroveň níž, tedy do jednotlivých fór Esetu. Kam které vlákno patří lze rozeznat podle cizího klíče.



id_postu	id_vlakna	text	kdo	hodnost	bydliste	datum_postu	datum_registrace
1	1	<MEMO>	Matthijs5nl	Frequent Poster	NEZADANO	2011-03-08 06:06:00	Jan 2010
2	1	<MEMO>	SweX	Very Frequent Poster	Sweden	2011-03-08 13:05:00	Apr 2007
3	1	<MEMO>	MinDokan	Infrequent Poster	NEZADANO	2011-03-08 21:05:00	Sep 2010
4	1	<MEMO>	The_ChamP	Frequent Poster	Mumbai	2011-03-09 01:23:00	Mar 2010
5	1	<MEMO>	dotxzer0	Infrequent Poster	NEZADANO	2011-03-09 02:24:00	Dec 2010
6	1	<MEMO>	x ZauX x	Regular Poster	NEZADANO	2011-03-09 08:40:00	May 2010
7	1	<MEMO>	troy1987	Infrequent Poster	NEZADANO	2011-03-09 08:46:00	Jun 2010
8	1	<MEMO>	Matthijs5nl	Frequent Poster	NEZADANO	2011-03-09 10:52:00	Jan 2010
9	1	<MEMO>	tipo	Frequent Poster	romania	2011-03-09 11:25:00	Dec 2008
10	1	<MEMO>	SweX	Very Frequent Poster	Sweden	2011-03-09 13:13:00	Apr 2007
11	1	<MEMO>	xxJackxx	Frequent Poster	NEZADANO	2011-03-09 13:33:00	Oct 2008
12	1	<MEMO>	The_ChamP	Frequent Poster	Mumbai	2011-03-09 13:36:00	Mar 2010
13	1	<MEMO>	SweX	Very Frequent Poster	Sweden	2011-03-09 15:20:00	Apr 2007
14	1	<MEMO>	CogitoTesting	Frequent Poster	Sea of Tranquility	2011-03-09 15:40:00	Jul 2009
15	1	<MEMO>	ExtremeGamerBR	Frequent Poster	NEZADANO	2011-03-09 15:41:00	Aug 2010
16	1	<MEMO>	dotxzer0	Infrequent Poster	NEZADANO	2011-03-09 22:16:00	Dec 2010
17	1	<MEMO>	yongsua	Regular Poster	Malaysia	2011-03-10 02:16:00	Feb 2011
18	1	<MEMO>	yongsua	Regular Poster	Malaysia	2011-03-10 02:25:00	Feb 2011
19	1	<MEMO>	vlk	AV Expert	NEZADANO	2011-03-10 03:23:00	Dec 2002
20	1	<MEMO>	Cutting_Edgetech	Frequent Poster	USA	2011-03-10 06:03:00	Mar 2006
21	1	<MEMO>	Cutting_Edgetech	Frequent Poster	USA	2011-03-10 06:12:00	Mar 2006
22	1	<MEMO>	cerberrr	Infrequent Poster	NEZADANO	2011-03-10 13:13:00	Jul 2010
23	1	<MEMO>	SweX	Very Frequent Poster	Sweden	2011-03-10 14:34:00	Apr 2007
24	1	<MEMO>	agoretsky	Eset Moderator	California	2011-03-10 21:54:00	Apr 2006
25	1	<MEMO>	The Hammer	Massive Poster	Toronto Canada	2011-03-10 22:18:00	May 2005
26	1	<MEMO>	Matthijs5nl	Frequent Poster	NEZADANO	2011-03-08 06:06:00	Jan 2010
27	1	<MEMO>	SweX	Very Frequent Poster	Sweden	2011-03-08 13:05:00	Apr 2007
28	1	<MEMO>	MinDokan	Infrequent Poster	NEZADANO	2011-03-08 21:05:00	Sep 2010
29	1	<MEMO>	The_ChamP	Frequent Poster	Mumbai	2011-03-09 01:23:00	Mar 2010
30	1	<MEMO>	dotxzer0	Infrequent Poster	NEZADANO	2011-03-09 02:24:00	Dec 2010
31	1	<MEMO>	x ZauX x	Regular Poster	NEZADANO	2011-03-09 08:40:00	May 2010
32	1	<MEMO>	troy1987	Infrequent Poster	NEZADANO	2011-03-09 08:46:00	Jun 2010
33	1	<MEMO>	Matthijs5nl	Frequent Poster	NEZADANO	2011-03-09 10:52:00	Jan 2010
34	1	<MEMO>	tipo	Frequent Poster	romania	2011-03-09 11:25:00	Dec 2008
35	1	<MEMO>	SweX	Very Frequent Poster	Sweden	2011-03-09 13:13:00	Apr 2007
36	1	<MEMO>	xxJackxx	Frequent Poster	NEZADANO	2011-03-09 13:33:00	Oct 2008
37	1	<MEMO>	The_ChamP	Frequent Poster	Mumbai	2011-03-09 13:36:00	Mar 2010
38	1	<MEMO>	SweX	Very Frequent Poster	Sweden	2011-03-09 15:20:00	Apr 2007
39	1	<MEMO>	CogitoTesting	Frequent Poster	Sea of Tranquility	2011-03-09 15:40:00	Jul 2009
40	1	<MEMO>	ExtremeGamerBR	Frequent Poster	NEZADANO	2011-03-09 15:41:00	Aug 2010
41	1	<MEMO>	dotxzer0	Infrequent Poster	NEZADANO	2011-03-09 22:16:00	Dec 2010
42	1	<MEMO>	yongsua	Regular Poster	Malaysia	2011-03-10 02:16:00	Feb 2011
43	1	<MEMO>	yongsua	Regular Poster	Malaysia	2011-03-10 02:25:00	Feb 2011
44	1	<MEMO>	vlk	AV Expert	NEZADANO	2011-03-10 03:23:00	Dec 2002
45	1	<MEMO>	Cutting_Edgetech	Frequent Poster	USA	2011-03-10 06:03:00	Mar 2006
46	1	<MEMO>	Cutting_Edgetech	Frequent Poster	USA	2011-03-10 06:12:00	Mar 2006
47	1	<MEMO>	cerberrr	Infrequent Poster	NEZADANO	2011-03-10 13:13:00	Jul 2010

Obrázek 8: Seznam jednotlivých postů vlákna

Obrázek 8 znázorňuje výsledek zanoření do poslední úrovně fóra. Zanoří se do každého vlákna a uloží jednotlivé příspěvky vlákna – posty neukládá se pouze text, ale i informace o autorovi a příspěvku jako celku. Příslušnost jednotlivých postů k jednotlivým vláknům jde rozeznat podle cizího klíče.

4.7 Problémy a omezení

Moje práce začínala seznamováním s vybraným fórem v té době jsem netušil, že Eset fórum je jen subfórem to vyšlo najevo až při bližším seznámení. Stejně tak jsem zjistil, že administrátoři tohoto fóra zablokovali member list což pro mě bylo celkem zklamání, jelikož právě z něj jsem chtěl nakonec práce udělat nejvíc statistických údajů. Po drobné konzultaci s vedoucím práce jsme se domluvili, že statistiku zaměříme na příspěvky, jejich četnost a jiné věci, které byly dostupné čímž jsme tuto záležitost uzavřeli a mohl jsem pokračovat dále.

Hned na začátku práce jsem narazil na problém se kterým jsem si nevěděl rady. Server na moje požadavky vracel kód 400 – Bad Request studoval jsem tento problém a došel jsem k závěru, že chyba v mém požadavku může být v čemkoliv.
Další komplikace – ale jen drobná nastala v momentě, kdy jsem začal stahovat ze sítě větší počet informací dostal jsem ban. Zjistil jsem, že špatně nastavuji proxy a budu muset přepracovat přístup k fóru.

Postupem času se objevil problém, který mi znemožnil stáhnout text příspěvku do databáze. Zásadní chyba byl v tom, že jsem si text ukládal do InputLine a ta se mi načetla vždy jen jeden řádek, takže když jsem použil regulání výraz vždy mi příspěvek osekal na jeden řádek a to ten první další řádky již nevyhovovaly a zahodil je.To mě přivedlo k

```
StringBuilder line = new StringBuilder();
line.append(inputLine);
```

což mi výrazně urychlilo práci z textem a v neposlední řadě mi umožnilo stáhnout celý příspěvek, proto poslední zanoření do sociální sítě se výrazně liší od předešlých zanoření. A musím říci, že kdybych tento projekt dělal znovu pro práci s textem bych asi použil StringBuilder mnohem častěji.

5. Graf soc.sítě

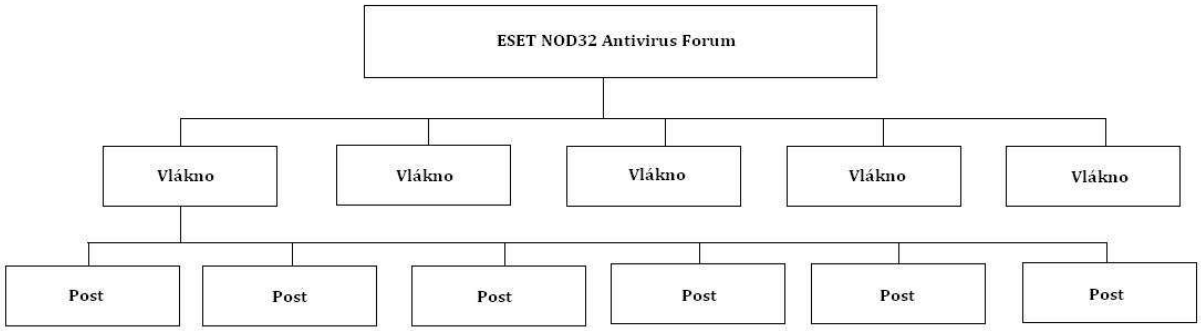
Abych ukázal graficky strukturu fóra sestrojím graf do kterého budu vkreslovat údaje z db. Jak je vidět z čísel z kapitoly 2.4 znázornit celé fórum by asi nemělo smysl kvůli přehlednosti a velikosti, ale budu se snažit sestrojit nejkomplexnější graf sociální sítě. Aby z grafu šlo něco vyčíst a přitom nezabral hodně místa, bude třeba vybrat ten nejvhodnější. Graf bude 2D k jeho vytvoření použiji opět javu a data z db.

5.1 Typy grafů

V práci se věnuji některým vybraným typům grafu vhodným na tento typ implementace, protože některé svou kostrou nejsou schopny dostát požadavkům této práce. Graf musí být přehledný a přiměřeně velký.

Jednoduchý graf množin pojmů

Jedním z nejzákladnějších a všeobecně použitelných vizualizačních nástrojů v dolování dat z textu je jednoduchá hierarchická stromová struktura. Na obrázku 9. vidíme klasickou vizualizaci pro klasifikaci pojmů v kolekci dokumentů. Kořen a list vrcholů (uzlů), takové vizualizace jsou jednotlivé identifikátory pojmů (např. jména pro označení pojmů). Tento druh vizualizačního nástroje může být také snadno uživateli přizpůsoben tak, aby mohl kliknout na uzel a posunout se směrem k základním dokumentům obsahujícím pojem. Nejběžnější způsob jak graficky zobrazit množinu pojmů je tedy právě pomocí jednoduché hierarchické stromové struktury. Uživatel může pracovat s tímto grafem vybráním uzlů, otevřít a uzavřít uzly, nebo definovat nové hledání s ohledem na tyto uzly například pro rozšíření stromu.[10]

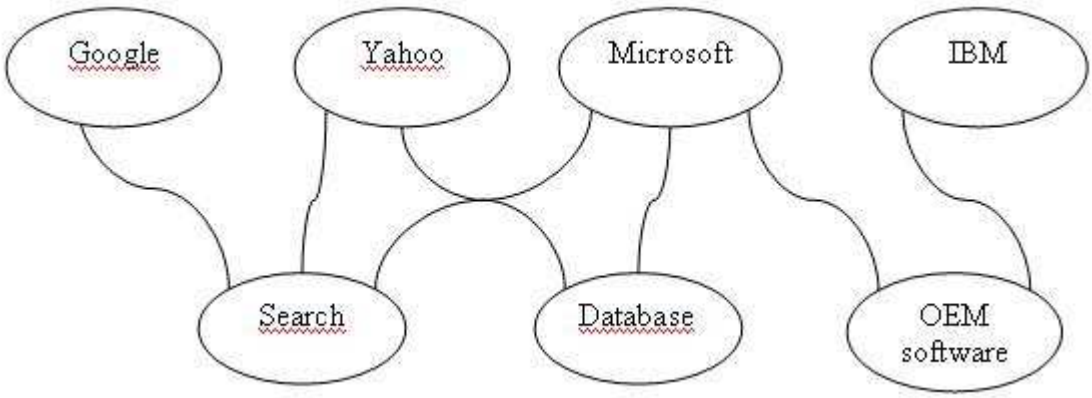


Obrázek 9: Jednoduchý hierarchický graf stromové struktury

Jenže takový graf pro naše zobrazení je nepoužitelný. Jak je vidět kořen je jedna sekce, ale už počet vláken by se nedal zobrazit celý a ani počet jednotlivých postů. Kdyby se vybral jen výřez stále by to nestačilo i přes své výhody tento graf pro mě není použitelný díky velikosti, které by nabral a v neposlední řadě i kvůli přehlednosti.

Jednoduchý graf asociací pojmů

Zaměřuje se na reprezentaci spojení - asociací. Je složen z jednotlivých vrcholů, které mohou být hranami připojeny k množině několika dalších vrcholů. Tento typ grafu je typicky používán ke spojení pojmů určité kategorie. V každém vrcholu takového grafu je vždy pouze jeden pojem. Dva pojmy jsou spojeny hranou, jestliže je jejich podobnost s ohledem na podobnostní funkci větší než dané omezení [10]

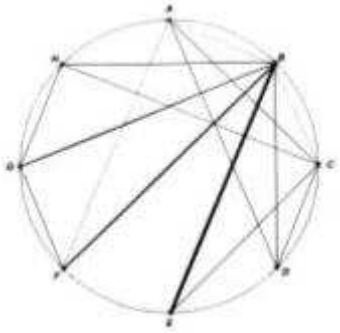


Obrázek 10: Graf asociací pojmů

Kdybych použil tento graf zobrazené informace by se ztratila v nepřehlednosti a velikosti grafu.Pro tyto zápory si ho nevyberu.

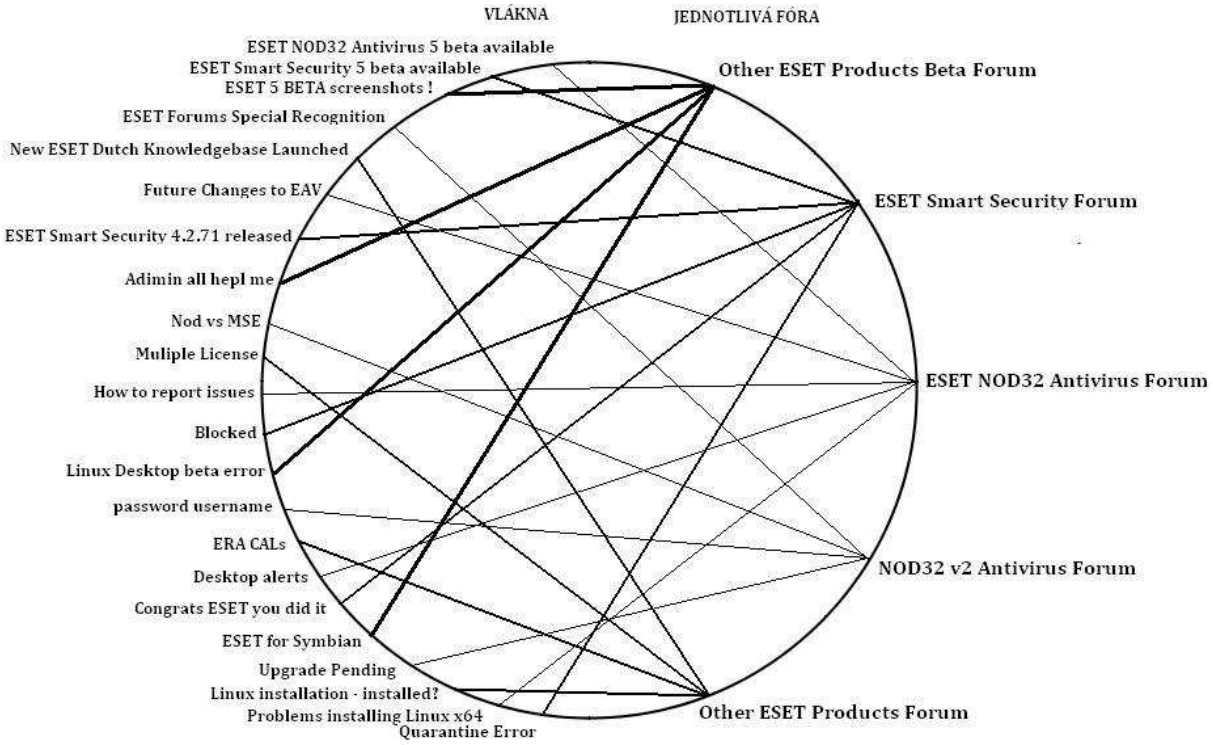
Kruhový graf

Kruhový graf je vizualizační metoda, která se dá použít pro umístění velkého množství informací do dvourozměrného formátu. Často se o něm hovoří jako o vizualizačním přístupu zřejmém „na první pohled“, protože není potřeba žádné další navigace, abychom dostali kompletní a velmi přesnou vizualizaci pro případný rozsáhlý objem dat.[10]



Obrázek 11: Ukázka kruhového grafu

Myslím si, že pro naše potřeby celkem vyhovuje. Ne pro zobrazení celého fóra, ale nějakého výřezu např.zobrazení hodiny nebo dne.



Obrázek 12: Jednotlivá fóra a jejich vlákna

Na obrázku 11. je zobrazen výřez fóra firmy ESET na pravé straně jsou zahrnuta všechna subfóra a na straně levé jsou vyfiltrovány jejich vlákna. Tento obrázek vznikl na základě informací které, nám crawler získal s fóra.Pro přehlednost jsem zvolil odlišnou tloušťku spojovacích linek.

6. Statistiky a zhodnocení

První věc kterou budu chtít zjistit o indexované sítí je jakou rychlostí na něm přibývají příspěvky.

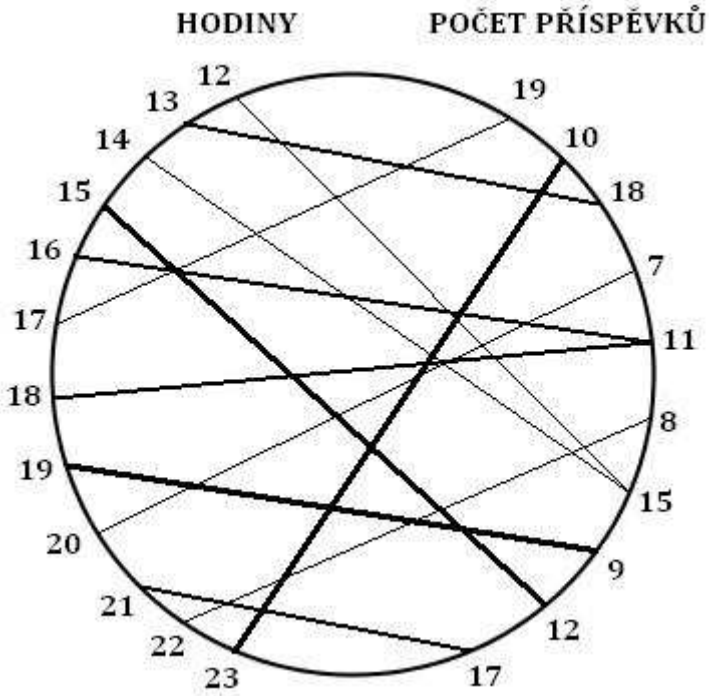
```
SELECT * FROM post WHERE SUBSTRING(datum_postu,12,2) = "16" GROUP BY datum_postu;
```

Výpis 33: Seznam příspěvku v jednotlivých hodinách

Na výpisu 33 je vidět SQL dotaz, který mi vrátí počet příspěvků, které byly napsány v určitou hodinu.

Hodina	Počet příspěvků
12	15
13	18
14	15
15	12
16	11
17	19
18	11
19	9
20	7
21	17
22	8
23	10

Tabulka 2: Počet příspěvků ve vybraných hodinách



Obrázek 13: Počet příspěvků za hodinu

Obrázek 13 je grafická podoba tabulky 2. Grafické znázornění počtu příspěvků za hodinu.

```
SELECT * FROM post WHERE SUBSTRING(datum_postu,1,10) = "2011-03-13" GROUP BY
datum_postu;
```

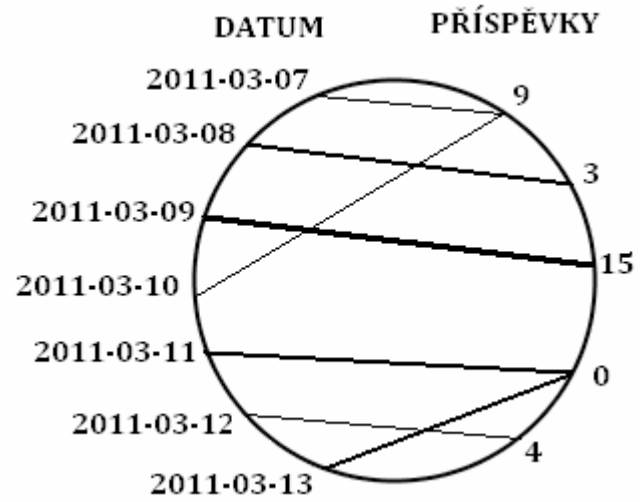
Výpis 34: Výpis příspěvku za daný den

Výpis 34 naznačuje SQL dotaz na databázi, který použiji abychom získali výpis příspěvků za jednotlivé dny. Jelikož datum_postu je formát datetime je třeba použít funkci substring, abych vybral jen tu část data kterou chci. Dále pak funkce group by zaručuje, že nám do seznamu nepronikne žádná duplicita.

Datum	Počet příspěvků
2011-03-07	9
2011-03-08	3
2011-03-09	15

2011-03-10	9
2011-03-11	0
2011-03-12	4
2011-03-13	0

Tabulka 3: Týdenní přírůst příspěvků



Obrázek 14: Týdenní přírůst příspěvků

Další věc, která mě bude zajímat je kolik který uživatel napsal.Náhodně vyberu sedm z nich.

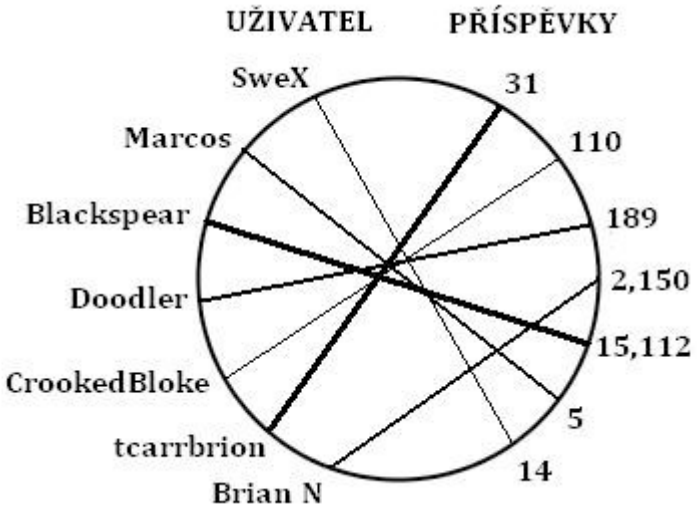
```
SELECT * FROM post WHERE kdo= "SweX" GROUP BY datum_postu;
```

Výpis 35: Seznam příspěvků daného autora

Ve výpisu 35. je vidět SQL dotaz na db, který nám vrátí seznam příspěvků daného autora. Záměrně v dotazu nepoužívám funkci count(), jelikož používám funkci group by tyto dvě funkce se vylučují. Funkce group by je pro mě důležitější, protože mě zbavuje případných duplicit.

Autor	**Počet příspěvků**
SweX	14
Marcos	5
Blackspear	15,112
Doodler	189
CrookedBloke	110
tcarrbrion	31
Brian N	2,150

Tabulka 4: Počet příspěvků jednotlivých uživatelů



Obrázek 15: Počet příspěvků daného autora

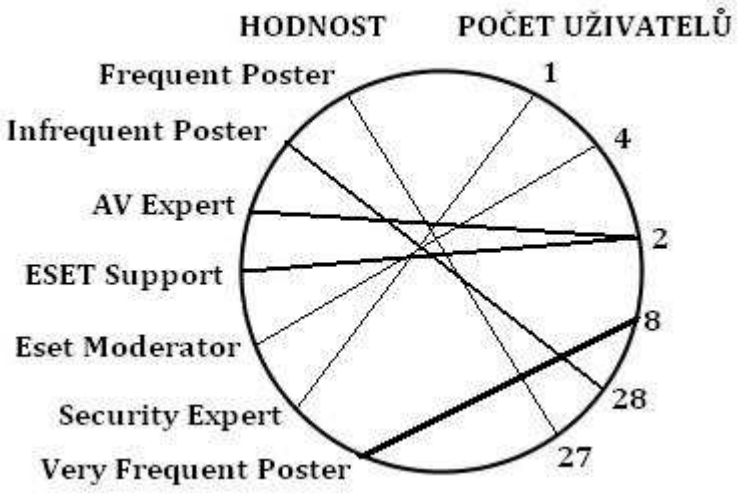
Na fóru jsou zavedeny různé hodnosti, podle toho kdo kolik poslal příspěvků.

```
SELECT * FROM post where hodnost = "Frequent Poster" GROUP BY kdo;
```

Výpis 36: Seznam uživatelů stejné hodnosti

Hodnost	Počet uživatelů
Frequent Poster	27
Infrequent Poster	28
Very Frequent Poster	8
ESET Support	2
Eset Moderator	4
Security Expert	1
AV Expert	2

Tabulka 5: Počet uživatelů majících stejnou hodnost



Obrázek 16: Počet uživatelů dané hodnosti

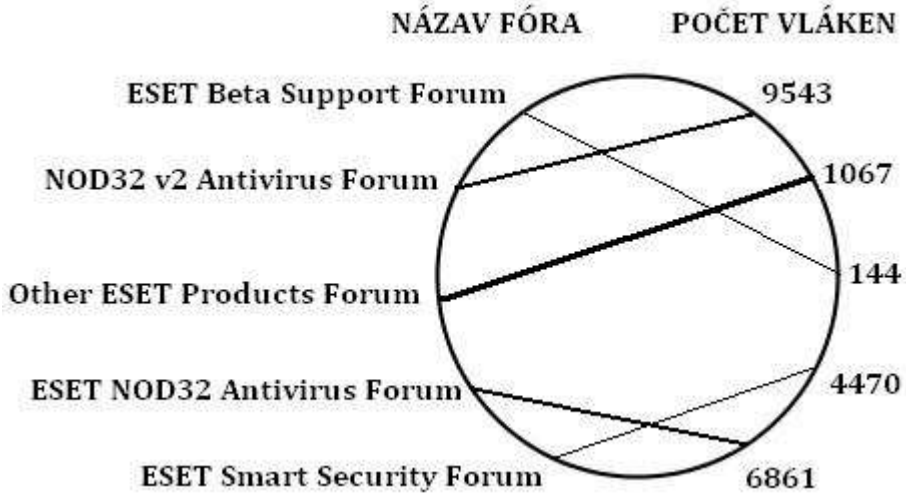
Statistiku je možno provádět nad celým fórem a ne jen nad příspěvky. Je možno zjistit kolik vláken mají jednotlivá subfóra Esetu.

```
SELECT nazev_fora,pocet_vlaken FROM forum;
```

Výpis 37: Počet vláken jednotlivých fór

Název Fóra	Počet vláken
ESET Beta Support Forum	144
NOD32 v2 Antivirus Forum	9543
Other ESET Products Forum	1067
ESET NOD32 Antivirus Forum	6861
ESET Smart Security Forum	4470

Tabulka 6: Jednotlivé fóra a počet jejich vláken



Obrázek 17: Jednotlivá fóra a počet jejich vláken

7. Zhodnocení a závěr

Cílem této závěrečné práce bylo sestrojení crawlera což se nám podařilo. Sestrojený crawler není úplně dokonalý, ale stačí na úkol pro který byl vytvořen. Zlepšení do budoucna bych viděl ve změně vyhledávacího algoritmu a zefektivnění vyhledávání v textu. Při samotném vyhledávání bych znovu použil regulární výrazy, jejich vlastnosti přesně odpovídají požadavkům pro práci s textem. Dále bych vylepšil procházení jednotlivých vláken. Tak aby nezabírala tolik času. Hodně času mi zabralo sestavení správného requestu na server, kdy jsem musel přijít na to jaký nebo spíš co všechno musí obsahovat dotaz na server. Vybral jsem si sociální síť a zindexoval ji - pomocí mého crawlera a na základě toho můžu dělat různé statistiky, kolekce vybírat si různé informace. Crawler bohužel není sestaven pro jakoukoli sociální síť, lze jej použít jen na fórum Esetu. Musím říci, že nedostatkem celého systému je ruční vyhledávání proxy a její rychlá spotřeba – admini udělují bany hodně rychle.

8. Literatura

[1] Adrian Bondy and U.S.R. Murty: *Graph Theory (Graduate Texts in Mathematics)*, 2007. 654 s. ISBN-10: 1846289696. ISBN-13: 978-1846289699.

[2] Stanley Wasserman and Katherine Faust, *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences),* 1994. 857 s. ISBN-10: 0521387078. ISBN-13: 978-0521387071.

[3] Ing.Peter Scherer, *Vizualizace v Information Retrieval pomocí Microsoft Silverlight,* Diplomová práce VŠB – TU Ostrava, 2009

[4] Petr Kopka, *Vizualizace v Information Retrieval*, Diplomová práce, VŠB-TU Ostrava, 2008

[5] URL http://cs.wikipedia.org/wiki/Soci%C3%A1ln%C3%AD_s%C3%AD%C5%A5 (20.11.2010)

[6] URL http://www.cs.vsb.cz/sawa/uti/ (30.4.2011)

[7] URL http://en.wikipedia.org/wiki/Web_crawler (2.5.2011)

[8] URL http://www.regular-expressions.info/examples.html (15.1.2011)

[9] URL http://haacked.com/archive/2004/10/25/usingregularexpressionstomatchhtml.aspx (20.1.2011)

[10] URL http://download.oracle.com/javase/tutorial/essential/regex/ (12.11.2011)

[11] URL http://programovani.blog.zive.cz/2009/08/java-jednoduchy-pristup-k-databazi-mysql/ (29.4.2011)

[12] URL http://download.oracle.com/javase/tutorial/essential/regex/ (20.3.2011)

[13] URL http://www.java2s.com/Code/Java/2D-Graphics-GUI/Drawline.htm (30.4.2011)